

OMNIMOD: Automating Ontology Modularization for Digital Library Data Using CIDOC-CRM as Use Case

Giulia Biagioni ^a

*Unit ISP, Netherlands Organisation for Applied Scientific Research (TNO), New Babylon,
Anna van Buerenplein 41a, Den Haag, The Netherlands*

Keywords: Ontology Modularization Method, Evaluation Method for Ontology Modularization, Cognitive Load Theory.

Abstract: This paper introduces OMNIMOD, a new method designed to modularize ontologies, RDF-based structures that organize knowledge within specialized domains. By simplifying complex information into manageable components, OMNIMOD enhances the analysis, understandability, and navigation of large ontological frameworks while also extending its functionality to include the modularization of associated data records, known as instance data. The method has been developed based on theoretical insights gathered from Cognitive Load Theory (CLT) and has been successfully tested and applied to CIDOC-CRM (Conceptual Reference Model of the International Committee for Documentation), the ISO standard for describing data related to cultural heritage materials. The accompanying Python functions, developed for OMNIMOD and provided in the corpus of the text, empower readers to adapt and utilize OMNIMOD according to their specific needs.


1 INTRODUCTION

Ontology modularization, which involves breaking down an ontology into smaller, more manageable parts, is particularly valuable for large ontologies (Parent and Spaccapietra, 2009). This technique improves reasoning capabilities (Gatens et al., 2014), simplifies ontology maintenance (Sarkar and Dong, 2011), enhances alignment with external ontologies (Ghafourian et al., 2013), assists in the structural examination of the taxonomy and axioms (Vescovo et al., 2011a), and allows for selective access to specific sections of the ontology while concealing others (Konev et al., 2009). As repositories of knowledge that enable meaningful communication between agents, ontologies are naturally designed to grow and incorporate an ever-expanding body of knowledge. This expansion is further motivated by European initiatives focused on establishing data spaces and digital libraries¹ grounded in ontologies, aiming to organize and utilize data more efficiently (Corcho and Simper, 2022; Commission, 2022).

Despite its recognized importance within the scientific community, the field of ontology modulariza-

tion faces significant hurdles, as outlined by LeClair and colleagues (LeClair et al., 2023). These challenges span from the absence of uniform guidelines for conducting the modularization process to the need for more intuitive methods for examining modularized entities (d'Aquin et al., 2009; LeClair et al., 2023). Additionally, there is also a lack of techniques that take into account the data records structured using the ontology in the modularization process, as well as standardized methods to evaluate the modularization process. Such obstacles markedly hinder the processes of navigation, analysis, and the assessment of intricate knowledge structures.

To address these issues, this study presents a new methodology for ontology modularization referred to here as OMNIMOD. The approach centers around CIDOC-CRM for the description of cultural heritage assets as primary use case. By doing this, the current initiative pays a double contribution. On the one hand, it aims to refine modularization techniques and enhance the management, understanding, and practical application of extensive ontologies. On the other hand, it seeks to provide cultural heritage professionals and users of extensive ontological conceptualizations with specifically designed tools for automated ontology modularization, enhancing their ability to access, comprehend, and work within large ontolog-

^a  <https://orcid.org/0000-0002-9005-7945>

¹An example of European digital library for cultural heritage material is provided by <https://www.europeana.eu/en>

ical frameworks.² The methodology behind the development of OMNIMOD is informed by insights from cognitive load theory (CLT).

CLT is a conceptual framework that describes how the human brain processes and retains new information. It offers strategies for organizing and presenting information in ways that enhance users' ability to assimilate new data effectively while fostering a deeper understanding of the material. By including principles of CLT in its design, OMNIMOD leverages achievements in the humanities to enhance its technical implementation and formulate the theoretical strategy that underpins its approach. The name "OMNIMOD," which stands for Ontology Modularization through an Integrated Methodological Design, reflects this comprehensive approach. The efficiency of OMNIMOD has been evaluated based on two criteria: cohesion and coupling. The results are presented in the paper.

This study is structured as follows: beyond the introduction already presented, section 2 delineates the state of the art, providing a critical examination of existing methodologies and identifying the need for advancements in ontology modularization. Section 3 discusses the methodology employed to develop OMNIMOD, which integrates both the theoretical frameworks that guided its design and the context for its creation. Finally, section 4 introduces OMNIMOD itself, detailing both its formal definition and technical implementation. The metrics used to evaluate the method are presented in section 5. The validation process of OMNIMOD, tested with CIDOC-CRM as use case, is outlined in Section 6. Ultimately, section 7 further discusses the results presented in Section 6.

²The modularization method presented in this study gains additional relevance in light of Directive 2013/37/UE. Enforced in 2021, the directive requires cultural heritage institutions to make their data machine-understandable, interconnected, and openly accessible in digital libraries. Adopting standardized, comprehensive ontologies like CIDOC-CRM presents a significant challenge for professionals tasked with navigating and adapting these broad ontological frameworks to publish their data on both digital platforms and digital libraries (such as Europeana). Therefore, the tools and methodologies developed in this study are not only designed to overcome the current limitations in ontology modularization but also to support cultural heritage professionals in fulfilling the mandates of Directive 2013/37/UE, ensuring they are provided with the possibility to access and understand how to efficiently and consistently employ extensive vocabularies.

2 STATE OF THE ART

Ontology modularization strategies can vary widely based on several factors. They might use semantic or syntactic criteria to determine how to divide the ontology into smaller, more manageable parts. The approach to modularization can be guided by either informal or formal specifications. In terms of implementation, these strategies range from fully automated processes, where software tools do the work, to partially automated methods that require some human intervention, and entirely manual approaches where the modularization is manually implemented. The objectives behind modularizing an ontology can also differ significantly, influencing the chosen strategy. For instance, the aim might be to enhance the ontology's usability, improve reasoning efficiency, or facilitate easier maintenance and updates. Another critical consideration is how the resulting modules interact with each other, whether they should be disjoint, meaning no overlap in concepts or properties, or if overlapping modules are permissible.

Methods that employ syntactic strategies are commonly referred to as graph-based approaches, focusing on the structural relationships and connections within the ontology represented as a graph. These approaches analyze the topology and connectivity of the ontology graph to identify modular structures. Examples of the application of these methods are provided by (Noy and Musen, 2009; Sarkar and Dong, 2011; Kachroudi et al., 2013). On the other hand, methods that utilize semantic criteria are known as logical approaches. These approaches consider the meaning and semantics of the ontology's concepts and relationships to modularize the ontology based on its content. Examples of how these methods have been applied can be drawn from (Vescovo et al., 2011b; Kontchakov et al., 2010; Grau et al., 2009). Ultimately, a third category, referred to as hybrid methods, is identified in (LeClair et al., 2023). These incorporate elements of both syntactic and semantic approaches by leveraging both the structural properties of the ontology graph and the semantic information encoded within it. An instance of this approach is exemplified in (Leclair et al., 2019).

According to insights from the field, there are several areas where existing ontology modularization methods could be significantly improved. A primary suggestion is to extend the focus beyond merely modularizing the ontology to include the associated data records (LeClair et al., 2023). This wider scope is expected to ensure a more exhaustive coverage of the domain described within the modules. Moreover, the establishment of clear, unambiguous criteria for eval-

uating the modularization process is emphasized as a critical need.

Therefore, to advance research in current ontology modularization methods, OMNIMOD has been developed to test the efficiency of and create a new modularization approach. This includes assessing its limitations, exploring potential alternative paths, and determining applicable evaluation criteria for the method and its scoped functionalities. The evaluation criteria to assess the modularization process of OMNIMOD are presented in this paper to further formalize and present a possible approach that can be followed to evaluate methods for ontology modularization. The methodology and theoretical underpinnings used to design OMNIMOD are detailed in the subsequent section.

3 METHODOLOGY

This section presents the methodology behind the development of OMNIMOD. Section 3.1 outlines the contextual factors that inspired the creation of OMNIMOD. Section 3.2 delves into the theoretical backbone of the methodology, harnessing insights from CLT to inform the design and structuring principles followed by OMNIMOD.

3.1 Context

To contribute to the advancement of the field, this study has implemented an automated approach for modularizing large ontologies into modules. This approach was developed within the framework of the research activities conducted under the Norm Engineering program, hosted by the Netherlands organization for applied scientific research (TNO). Within this program, specific ontologies have been created to facilitate the interpretation of norms, such as FLINT (Breteler et al., 2023), and Source ontology³.

The Source ontology, designed to manage textual information found in both digital and physical libraries regarding norms and regulations, aligns with key standards in the cultural heritage sector, including CIDOC-CRM⁴, and some of its extensions (i.e. the

³FLINT and Source ontologies can be accessed at <https://gitlab.com/normativesystems>

⁴CIDOC-CRM serves as the standard vocabulary for structuring data related to the collections of cultural heritage institutions. Initially developed in 1994 by an international committee of experts convened under ICOM, the model was technically expressed in Telos. Since then, it has evolved into various formats, including RDF(S) and OWL standards. Recognized as an ISO standard since 2009, the

Functional Requirements for Bibliographic Records, FRBR,⁵ and FRBR's lightweight variant, LRM)⁶.

To align the Source ontology with the standardized conceptual frameworks used within (digital) libraries, an in-depth investigation into CIDOC-CRM was conducted, inter alia. This process required breaking down the ontology into modules to analyze each class definition and its relationships in order to gain deeper insights into the conceptualizations and how they complement each other. Given the scarcity of existing automated methods suitable for modularizing ontologies and knowledge graphs, and facilitating in-depth examination of conceptualizations, OMNIMOD was developed. The development of this method has been steered by foundational principles of ontological knowledge organization, with a particular emphasis on class definitions, hierarchical structures, and class relationships. Therefore, while the method considers the semantic scope of each module, its core approach remains graph-based.

3.2 Theoretical Framework for the Modularization Strategy

OMNIMOD was developed with the aim to improve navigability, information retrieval, analysis and examination of extensive ontologies. On a theoretical level, the creation of modularization method was informed by findings from the field of CLT, namely, a theoretical framework that examines how information is processed within recipients' working memory and derives ways to present and organize information to maximize its uptake. Developed by John Sweller in the 1980s, CLT suggests that our ability to process and integrate data is influenced by the capacity of our working memory, which can be overloaded if too much information is presented at once (Sweller et al., 1998; Sweller et al., 2019). The theory divides cognitive load into three main types: intrinsic, extraneous, and germane.

Intrinsic load is the inherent difficulty associated with a specific topic. It is directly related to the com-

model serves as a fundamental framework within digital European libraries like Europeana, with Europeana EDM aligning seamlessly with CIDOC-CRM.

⁵FRBR, developed by the International Federation of Library Associations (IFLA), establishes the Work Expression Manifestation and Item (WEMI) Structure as its foundation. This framework distinguishes between abstract Work, its various Expressions, physical Manifestations embodying those Expressions, and individual Items specific to Manifestations.

⁶In February 2023, FRBR released the draft model of its lightweight WEMI conceptualization, LRM, maintaining alignment with CIDOC-CRM.

plexity of the material and the recipient's existing knowledge (Sweller et al., 1998). As such, intrinsic load cannot be altered by information organizational design, but understanding its impact is crucial for pacing and structuring information uptake.

Extraneous load, on the other hand, refers to the cognitive load imposed by the way information is presented to recipients (Sweller et al., 1998; Whelan, 2007). It is not essential for information uptake and can be reduced or managed through effective instructional design. Extraneous load can arise from disorganized and inadequately presented information that distracts or confuses recipients, requiring them to expend effort on processing irrelevant information. Reducing extraneous load is crucial for liberating working memory capacity to handle intrinsic and germane loads.

Germane load is the cognitive effort dedicated to processing, constructing, and automating schemas (Sweller et al., 1998). It represents the cognitive load that contributes directly to learning by facilitating the integration of new information into long-term memory. Effective organization and presentation of knowledge aim to maximize germane load by using strategies that support schema acquisition

The application of CLT has been significant in informing strategies to reduce extraneous load while maximizing the potential to assimilate information (Skulmowski and Xu, 2022). Research in multimedia learning has highlighted several principles aimed at minimizing extraneous load, thereby optimizing cognitive capacity for processing intrinsic content. Notable principles include the *split-attention effect*, advocating for the placement of related information in proximity to each other, and guidelines against including distracting or irrelevant details in information presented to end-users to avoid the *seductive details effect*. This effect can hinder recipients' understanding of core data by distracting them with irrelevant information presented alongside the main material.

When applied to ontology modularization methods and techniques, these principles may outline the strategy or approach that the modularization method should follow. Moreover, they can provide insights into which criteria should be primarily considered to evaluate the results of the modularization approach and, consequently, the approach itself. The focus points thus become cohesion of the information presented and grouped in each module, consistency of the information covered in each module, completeness of each module, domain appropriateness, integration (i.e. ease of integrating modules back into the original ontology while ensuring the degree of independence of each module), and the logical consistency

of the modules.

Translating such CLT principles into guidelines, the following points can be derived for the development approach of the modularization method. The method should create thematically cohesive modules, aiming to remove all superfluous information while still maintaining the richness of the axiomatization of the entities in the modules. As a result, cohesion, which refers to the degree to which the elements within a module are related to each other, and coupling, which refers to the degree of interdependence between modules, become primary criteria for the evaluation of the modularization approach.

In line with these principles, OMNIMOD has been designed to adhere to the guidelines derived from CLT, as will be presented in the following sections. Its efficiency is evaluated primarily in terms of coupling and cohesiveness, providing insights into its performance and highlighting space for potential improvements.

4 OMNIMOD

To effectively apply and implement the principles of CLT in the creation of ontological modules, it is crucial to first comprehend the foundational organization of information in OWL (Web Ontology Language) and RDF(S) (Resource Description Framework Schema), or more broadly, the underlying philosophy that guides the technical implementation of these two languages⁷.

RDF and OWL semantics are built on the fundamental distinction between the concepts of universality (`rdfs:resource` and `owl:Thing`) and nullity (`owl:Nothing`). The classes `rdfs:resource` and `owl:Thing` serve as universal or "upper level" classes, signifying that all instantiable classes fall under them. Conversely, the concept of the empty set is captured in OWL with `owl:Nothing`, and in RDF, a similar concept is implied though not explicitly named, highlighting the framework's capability to represent the absence of any class instances.

By adhering to this framework, semantic technologies inherently propose that all entities trace back to a foundational origin situated in a root class, while entities that cannot be instantiated are assigned to an empty set. This dichotomy serves as the basis for organizing knowledge within these systems. Subsequently, classes receive further characterization through the properties (or relationships) they form

⁷For RDFs and OWL documentation cf. respectively: <https://www.w3.org/TR/rdf-schema/> and <https://www.w3.org/TR/owl-guide/>

with other groups of individuals. These properties not only refine but also augment the definitions of classes. The nature of the relationships between individuals dictates their classification, determining whether they belong to one class or another.

OWL class restrictions, along with RDFS domain and range definitions, significantly augment the axiomatic description of classes. More specifically, OWL Class Restrictions provide a powerful means to define conditions that individuals must satisfy to belong to a particular class. These restrictions can specify necessary and sufficient conditions for class membership based on the presence or absence of certain properties and their values. On the other hand, RDFS domain and RDFS range definitions indirectly influence class membership by asserting that individuals with certain properties belong to particular classes or that the values of specific entities characterized through the use of properties must fit within defined categories.

Summarily, the above described organizational logic follows a class-centric approach where sets and groups of individuals are categorized into specific classes based on the attributes and relationships they exhibit. Subclass relationships, in particular, establish a hierarchical structure among classes, implying that properties and attributes are inherited and shared down the hierarchy from upper classes to their subclasses.

The method proposed in this study respect the aforementioned class-centric approach to modularize knowledge representation. This means that it breaks down complex hierarchical structures into distinct, manageable modules based on the organization and definition of the classes. Each module not only includes its own set of properties (where the classes act as either the domain or the range) but also encompasses the individuals (data records) and the relationships these individuals instantiate. The formal definition of the modularization method used to develop OMNIMOD can be described as presented below, beginning with the notation utilized and a description of this notation, as detailed in Table 1.

The modularization method followed by OMNIMOD aims to break the ontology into distinct modules based on the direct subclasses of the root class R . Each module is structured as follows:

- The root class R and its definitions.
- Each direct subclass C within $\text{Sub}(R)$, including their complete subclass hierarchy ($\text{Sub}^*(C)$).
- Definitions, restrictions, and properties associated with R , C , and subclasses within $\text{Sub}^*(C)$.
- All individuals that instantiate R , C , or any sub-

Table 1: Notation Overview.

Notation	Description
O	Ontology.
C	The set of all classes within O .
\mathcal{P}	All properties in O .
I	All individuals in O .
$\text{Sub}(C)$	The set of direct subclasses of any class C in O , used specifically for the root class R .
$\text{Sub}^*(C)$	All subclasses (direct and indirect) under each direct subclass C within $\text{Sub}(R)$.
$\text{Def}(C)$	The definition of class C , including OWL restrictions and class-specific properties.
$\text{Dom}(P)$	The domain of property P .
$\text{Range}(P)$	The range of property P .
$\text{Triples}(I)$	RDF triples associated with individual I .
R	The root class for modularization purposes.

class within $\text{Sub}^*(C)$, along with the RDF triples involving these individuals.

- Properties where R , C , or any subclasses within $\text{Sub}^*(C)$ are the domain or range.
- Properties where the domain or range is not specified, and therefore potentially applicable to all classes.

The modularization approach implemented in OMNIMOD can be formally described as presented in Table 2.

Table 2: Overview of the OMNIMOD Method's Modular Structure.

Category	Modularization Method
Classes:	$C_M = \{R\} \cup \{C\} \cup \text{Sub}^*(C)$
Properties:	$\mathcal{P}_M = \{P \in \mathcal{P} : ((\text{Dom}(P) \in C_M \vee \text{Range}(P) \in C_M) \vee (\text{Dom}(P) = \emptyset \vee \text{Range}(P) = \emptyset))\}$
Instances:	$I_M = \{I \in I : \text{InstanceOf}(I, X), X \in C_M\}$ $\text{Triples}_M = \{\text{triples involving } I : I \in I_M\}$

The modularization method delineated above is technically implemented as a Python function, making it accessible for anyone to try, reuse, and adapt as needed. This function, which is fully detailed below, requires two inputs: the ontology in question, and the root class from which the user wants to initiate the modularization process. The design of this function provides a degree of adaptability for subsequent in-depth analysis. Moreover, it outputs the modules in a Turtle (.ttl) file format, thus ensuring broad compatibility and facilitating further utilization within the

semantic web framework.

```

import rdflib
from rdflib import Graph, URIRef, BNode
from rdflib.namespace import RDF, RDFS, OWL

def get_local_name(uri):
    return uri.split('/')[-1].split('#')[-1]

def OMNIMOD(ontology_file, root_class_uri):
    g = rdflib.Graph()
    g.parse(ontology_file, format=rdflib.util.guess_format(ontology_file))

    # Ensure root_class_uri is a URIRef
    if isinstance(root_class_uri, str):
        root_class_uri = URIRef(root_class_uri)

    # Find immediate subclasses of the root class
    immediate_subclasses = set(g.subjects(RDFS.subClassOf, root_class_uri))
    all_properties = set(g.subjects(RDF.type, RDF.Property)) | set(g.subjects(RDF.type, OWL.ObjectProperty)) | set(g.subjects(RDF.type, OWL.DatatypeProperty))

    def recursively_add_triples(subject, module_graph):
        """
        Recursively add triples; specially handle blank nodes to capture all related triples.
        """
        for s, p, o in g.triples((subject, None, None)):
            module_graph.add((s, p, o))
            if isinstance(o, BNode):
                recursively_add_triples(o, module_graph)

    def add_class_hierarchy_and_properties(class_uri, module_graph):
        # Handle class hierarchy
        for s, p, o in g.triples((None, RDFS.subClassOf, class_uri)):
            module_graph.add((s, p, o))
            add_class_hierarchy_and_properties(s, module_graph)

        # Include all direct triples and recursively handled blank nodes linked via complex properties
        recursively_add_triples(class_uri,

```

```

module_graph)

# Handle properties where the class is the domain or range, or no domain/range specified
handled_properties = set()
for prop in set(g.subjects(RDFS.domain, class_uri)) | set(g.subjects(RDFS.range, class_uri)):
    handled_properties.add(prop)
    recursively_add_triples(prop, module_graph)

# Include properties with no domain or range in each module
for prop in all_properties - handled_properties:
    if not list(g.objects(prop, RDFS.domain)) and not list(g.objects(prop, RDFS.range)):
        recursively_add_triples(prop, module_graph)

# Include individuals that are instances of the class
for individual in g.subjects(RDF.type, class_uri):
    for s, p, o in g.triples((individual, None, None)):
        module_graph.add((s, p, o))

for subclass in immediate_subclasses:
    module_graph = Graph()
    add_class_hierarchy_and_properties(subclass, module_graph)
    filename = f"{get_local_name(subclass)}_module.ttl"
    module_graph.serialize(destination=filename, format='turtle')
    print(f"Module created: {filename}")

```

The code displayed above consists of several key parts. First, the necessary RDFLib modules are imported to handle RDF data. Then, the utility function *get local name* is provided to extract the local name from a URI. The main function, *OMNIMOD*, parses the ontology file and identifies immediate subclasses of the specified root class. It defines helper functions to recursively add triples and handle class hierarchies and properties, and creates separate modules for each subclass by capturing all relevant triples and properties.

OMNIMOD can be used by providing the path to the ontology that needs to be modularized and the root class from which the modularization process may start, as in the following example of usage ⁸.

⁸please note that OMNIMOD segments the ontology

```
# Example usage
OMNIMOD('CIDOC_CRM_v7.1.3.rdf', 'http://www.
cidoc-crm.org/cidoc-crm/E1_CRM_Entity')
```

5 EVALUATION METRICS

OMNIMOD has undergone testing and validation by applying it to CIDOC-CRM ontology⁹ with and without instances (i.e., data records). To evaluate OMNIMOD's efficacy, and therefore the effectiveness of the modularization method, measures for cohesion and coupling have been derived.

5.1 Cohesion and Coupling

Cohesion and coupling metrics evaluate the strength of relationships within and between modules (Khan, 2016). Cohesion refers to the relatedness of elements within a single module (García et al., 2010) (Oh et al., 2011). It is inferred from the intra-module relationship density. Higher density indicates higher cohesion, suggesting well-connected classes within the module. Coupling measures the interdependencies between different modules (Oh and Ahn, 2009). Lower coupling values indicate fewer dependencies, which is desirable for maintainability and scalability.

High cohesion and low coupling, as measured by average intra-module and inter-module relationship densities, align with CLT by reducing extraneous cognitive load and minimizing the split-attention effect.

5.1.1 Cohesion

Cohesion in the context of ontology modularization measures how well-connected the classes within a module are. The density formula (1) provides a way to quantify this by considering the number of logical axioms and the number of instantiated classes within the module. Logical axioms represent the relationships between classes, so a higher number of axioms generally indicates better connectivity.

Cohesion was calculated using the following metric for density in modules that had more than two instantiated classes:

into modules starting from a specified root class down through its branches. If users wish to start the modularization from all upper root classes, they should begin by indicating OWL:Thing or rdfs:Resource as the starting point for modularization. If the root classes are not explicitly instantiated as subclasses of these, users need to add such definitions to modularize from the upper part of the ontology.

⁹The version used is: v.7.1.3. It can be downloaded here: <https://www.cidoc-crm.org/versions-of-the-cidoc-crm>

$$\text{Density} = \frac{\text{Number of Logical Axioms}}{\text{Number of Classes} \times (\text{Number of Classes} - 1)} \quad (1)$$

In formula (1), the terms are defined as follows:

- **Number of Logical Axioms:** This is the total count of logical relationships between classes (such as subclass, equivalent class, property restrictions, object properties) defined within the module.
- **Number of Classes:** This is the total count of instantiated distinct classes present within the module. Note that this does not include distinct classes that are not instantiated and are derived from the domain and range external to the module.
- **(Number of Classes - 1):** This accounts for the potential relationships between the instantiated classes. Subtracting 1 ensures the density measure considers the proportion of actual relationships relative to the maximum possible relationships between different classes, normalizing the value based on the size of the module.

For modules including fewer than 2 instantiated classes, the strength of relation for each entity is calculated based on the farness centrality measure from graph theory, as proposed by (Freeman, 1978) and summarized in (Khan, 2016)

The results have been evaluated against the benchmark displayed in table 3:

Table 3: Benchmark to assess cohesion levels.

Cohesion Level	Cohesion Range
High Cohesion	≥ 0.5
Moderate Cohesion	0.2 \leq Normalized Cohesion $<$ 0.5
Low Cohesion	< 0.2

5.1.2 Coupling

Coupling was derived by calculating the ratio between A) the number of external classes with which the instantiated classes within the module established one or more relationships, and B) the number of instantiated classes within the module. The results were then normalized to provide an indication measure.

The following benchmark was used to assess coupling:

Table 4: Benchmarks for Coupling Levels.

Coupling Level	Coupling Range
High Coupling	≥ 0.5
Moderate Coupling	0.2 $<$ Average coupling value $<$ 0.5
Low Coupling	< 0.2

High coupling indicates strong dependencies between modules, which can be problematic as changes in one module might impact others. Conversely, low coupling suggests that the modules are relatively independent, which is desirable for maintainability and scalability.

6 RESULTS

OMNIMOD and its effectiveness have been tested on CIDOC-CRM, starting the modularization process from the "Entity" class. The resulting modules were named after the respective upper subclass under the CIDOC-CRM Entity from which they were derived. The modules created using OMNIMOD are as follows, each stored in a separate TTL file:

- E53 Place Module: Contains geographical data related to various locations.
- E54 Dimension Module: Includes measurements data.
- E92 Spacetime Volume Module: Integrates spatial and temporal data.
- E77 Persistent Item Module: Assists in managing artifact catalogues.
- E52 Time-Span Module: Provides detailed chronological information.
- E2 Temporal Entity Module: Focuses on the timing aspects of events.

The modules were evaluated based on the metrics for computing cohesion and coupling presented in the previous section. The average measure for cohesion across modules was 0.6, while that for coupling was 0.5. When tests were conducted with instance data, the data were correctly mapped within the module of the classes in which they were instantiated. Based on a comparative analysis across the modules and the core module, no information regarding properties and classes under the branches of the provided root class was lost. Logical consistency was maintained, as verified by running the Hermit 1.4.3.456 reasoner on each generated module, which revealed no contradictions. The inference was limited to the instantiated classes

and their relative relations, without extending to the classes that had no instantiation but were still present in the module due to being part of object properties' axiomatic definitions derived from the use of domain and range. Their inclusion allowed for integration of the module back into the core ontology, thereby meeting the requirement for evaluating possible reintegration into the main ontology as well.

7 DISCUSSION

OMNIMOD, and consequently its theoretical modularization approach, has scored highly in both cohesion and coupling. While the high cohesion score indicates that the richness of the axiomatization within each module is well-maintained, the high coupling value reveals potential scalability and maintainability issues. Specifically, if modifications are made within a module rather than in the core ontology, the high coupling may lead to difficulties in maintaining the integrity and consistency of the entire ontology.

These insights also prompt a point of reflection and help in better scoping the modularization approach. In fact, OMNIMOD can be a valuable support tool for users to understand, inspect, and analyze large ontological frameworks, as indicated by the cohesion score. Each generated module is self-contained and focuses on a specific thematic area, functioning as a standalone database. This possibly makes it easier to conduct targeted academic and professional exploration. Each module can show the main hierarchy and relationships of the branches of the root class provided as input to start the modularization, and also allow for the inclusion of instance data and their triples in the core module where their class is instantiated.

Moreover, by instantiating and providing axiomatization only for the classes that are the focal point of the module, while presenting the classes at the level of range and domain simply as such without further instantiation, the CLT-described split-attention effect and seductive details effect are potentially minimized. This approach aims to enable users to focus directly on data pertinent to their field without distractions. The results can span from improved user engagement through organized and easily navigable modules, to enhancing efficiency and accuracy in data retrieval and analysis. Ultimately, OMNIMOD may represent a valuable support for professionals seeking familiarity with ontologies to structure their data more effectively. It can certainly assist professionals in evaluating an ontology's expressivity coverage, enabling them to align their conceptualizations with established standards and potentially

extend these knowledge representations.

However, as previously addressed, from a technical perspective, OMNIMOD segments ontologies and instance data by creating duplicates. An instance in a triple within one module might also appear in a triple in another module. This principle applies to relationships and classes as well. Although OMNIMOD's duplication strategy may streamline exploration, it may also create issues if changes are made in one module without updating another module. Therefore, when changes are made, they should take place at the level of the integrated ontology rather than at the module level to avoid intensifying further work and creating possible problems.

The limitation described above also highlights key areas for further development. If readers of this paper wish to use and expand the core function provided in section 4, they should consider enabling OMNIMOD for users who want to directly work on the modules and have automated updates applied to all other modules. Specifically, a strategy for the automated updating of all modules when a change is made in one of them needs to be implemented. This enhancement would ensure consistency and integrity across the entire ontology, facilitating easier maintenance.

ACKNOWLEDGEMENTS

This research activity was conducted as part of the Norm Engineering Program at TNO, funded by the Dutch Ministry of the Interior and Kingdom Relations.

REFERENCES

- Breteler, J., van Gessel, T., Biagioni, G., and van Doesburg, R. (2023). The flint ontology: An actor-based model of legal relations. In *SEmantics2023:Proceedings*.
- Commission, E. (2022). Moving towards a european data space: New eu law for data-sharing. <https://data.europa.eu/en/news-events/news/moving-towards-european-data-space-new-eu-law-data-sharing>, last accessed 2024/04/18.
- Corcho, O. and Simper, E. (2022). The role of data.europa.eu in the context of european common data spaces. https://data.europa.eu/sites/default/files/report/data.europa.eu_theRoleofdata.europa.euinthecontextofEuropeancommondataspaces.pdf, last accessed 2024/04/18.
- d'Aquin, M., Schlicht, A., Stuckenschmidt, H., and Sabou, M. (2009). Criteria and evaluation for ontology modularization techniques. In *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, pages 67–89. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239.
- García, J., Peñalvo, F. J. G., and Therón, R. (2010). A survey on ontology metrics. In Lytras, M. D., de Pablos, P. O., Ziderman, A., Roulstone, A., Maurer, H. A., and Imber, J. B., editors, *Third World Summit on the Knowledge Society, WSKS'10*, volume 111 of *Communications in Computer and Information Science*, pages 22–27, Corfu, Greece. Springer. September 22–24.
- Gatens, W., Konev, B., and Wolter, F. (2014). Lower and upper approximations for depleting modules of description logic ontologies. *Frontiers in Artificial Intelligence and Applications*, 263:345–350.
- Ghafourian, S., Rezaeian, A., and Naghibzadeh, M. (2013). Graph-based partitioning of ontology with semantic similarity. In *Proceedings of the 3rd International Conference on Computer and Knowledge Engineering, ICCKE 2013*.
- Grau, B. C., Horrocks, I., Kazakov, Y., and Sattler, U. (2009). Extracting modules from ontologies: A logic-based approach. In *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, pages 159–186. Springer, Berlin, Heidelberg.
- Kachroudi, M., Zghal, S., and Yahia, S. B. (2013). Ontopart: At the cross-roads of ontology partitioning and scalable ontology alignment systems. *International Journal of Metadata, Semantics and Ontologies*, 8:215–225.
- Khan, Z. C. (2016). Evaluation metrics in ontology modules. *Description Logics*.
- Konev, B., Walther, D., and Wolter, F. (2009). Forgetting and uniform interpolation in extensions of the description logic el. In *CEUR Workshop Proceedings*, volume 477.
- Kontchakov, R., Wolter, F., and Zakharyashev, M. (2010). Logic-based ontology comparison and module extraction, with an application to dl-lite. *Artif. Intell.*, 174(15):1093–1141.
- Leclair, A., Khédri, R., and Marinache, A. (2019). Toward measuring knowledge loss due to ontology modularization. In *Scitepress*.
- LeClair, A., Marinache, A., Ghalayini, H. E., MacCaull, W., and Khedri, R. (2023). A review on ontology modularization techniques - a multi-dimensional perspective. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4376–4394.
- Noy, N. F. and Musen, M. A. (2009). Traversing ontologies to extract views. In *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, pages 245–260. Springer, Berlin, Heidelberg.
- Oh, S. and Ahn, J. (2009). Ontology module metrics. In *International Conference on e-Business Engineering (ICEBE'09)*, pages 11–18, Macau, China. IEEE Computer Society.
- Oh, S., Yeom, H. Y., and Ahn, J. (2011). Cohesion and coupling metrics for ontology modules. *Information Technology and Management*, 12(2):81–96.

- Parent, C. and Spaccapietra, S. (2009). An overview of modularity. In *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, pages 5–23. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Sarkar, S. and Dong, A. (2011). Characterizing modularity, hierarchy and module interfacing in complex design systems. In *Proceedings of the ASME Design Engineering Technical Conference*, volume 9.
- Skulmowski, A. and Xu, K. M. (2022). Understanding cognitive load in digital and online learning: A new perspective on extraneous cognitive load. *Educational Psychology Review*, 34:171–196.
- Sweller, J., van Merriënboer, J. J. G., and Paas, F. (2019). Cognitive architecture and instructional design: 20 years later. *Educational Psychology Review*, 31:261–292.
- Sweller, J., van Merriënboer, J. J. G., and Paas, F. G. W. C. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, 10:251–296.
- Vescovo, C. D., Gessler, D. D., Klinov, P., Parsia, B., Sattler, U., Schneider, T., and Winget, A. (2011a). Decomposition and modular structure of bioportal ontologies. In *The Semantic Web-ISWC 2011*, pages 130–145, United States. Springer Nature.
- Vescovo, C. D., Parsia, B., Sattler, U., and Schneider, T. (2011b). The modular structure of an ontology: Atomic decomposition and module count. *Frontiers in Artificial Intelligence and Applications*, 230.
- Whelan, R. R. (2007). Neuroimaging of cognitive load in instructional multimedia. *Educational Research Review*, 2(1):1–12.

